UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/266,207	03/10/1999	PAUL ENGLAND	777.215US1	5470

22801 7590 02/09/2005

LEE & HAYES PLLC
421 W RIVERSIDE AVENUE SUITE 500
SPOKANE, WA 99201

EXAMINER	
KLIMACH, PAULA W	

ART UNIT	PAPER NUMBER
2135	

DATE MAILED: 02/09/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/266,207	ENGLAND ET AL.	
	Examiner	Art Unit	
	Paula W Klimach	2135	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *19 October 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-77* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-77* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *07/21/04; 10/19/04*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Response to Amendment

This office action is in response to amendment filed on 10/19/04. Original application contained Claims 1-77. The amendment filed on 10/19/2004 have been entered and made of record. Therefore, presently pending claims are 1-77.

Response to Arguments

Applicant's arguments filed 10/19/04 have been fully considered but they are not persuasive because of following reasons.

Applicant argued, "... the Office Action fails to identify what publication or patent is meant by Arbaugh..." This is not found persuasive. In the PTO-892 that was sent with the office action 01/29/2004, Arbaugh is cited as an article in the section detailing non-patent literature.

Applicant argued, "...neither Barr nor Barlow is available as prior art under 35 U.C.C. 103 with respect to this application...". This is not found persuasive. The examiner points out that this application was filed on 03/10/1999 and to qualify for the change in 35 U.S.C 103 (c) the application would have to be filed after November 29, 1999 as indicated in the applicant's arguments of 10/19/2004.

The applicant argued further, "Arbaugh does not teach or disclose computing a cryptographic function of at least a portion of the operating system." This is not found persuasive. Arbaugh states "Finally the kernel (operating system) is verified by the last boot block in the chain before passing control to it," in section 3.2.2 paragraph 4 and in addition in

section 4 paragraph 2 Arbaugh states "... the verification function performs a cryptographic hash (cryptographic function)."

The applicant argued further, "Angelo teaches away from the disclosure of Arbaugh," stating as evidence Angelo "as opposed to relying solely on power-up routines to maintain a secure and trusted path." This is not persuasive. Angelo adds on top of the power-up routines by not relying solely on power-up routines, Angelo modifies the power-up routines by adding to them. Claims 1, 3, 11, and 19 are silent about "relying solely on power-up routines" or adding functionality to the system. In addition the combination of Angelo and Arbaugh teaches using the memory as in Angelo during the power-up routine as in Arbaugh. In fact Arbaugh discloses modification of the current bootstrap process by adding the verification step (3.2.2).

In reference to claims 3, 11, and 19 the applicant argued that Angelo is silent regarding the atomic operation. This is not found persuasive. Although Angelo and Arbaugh do not use the term atomic process or uninterruptible operation, the atomic process, in claims 3 and 11, is the cryptographic function performed on the operating system. Arbaugh discloses performing the cryptographic function (verification) in the Bootstrap code, therefore before the operating system is loaded, as a result it is an "uninterruptible operation"

In reference to claim 19, the applicant disagrees that Angelo discloses a software identity register. This is not found persuasive. The application does not define a software identity register, as a result, the software identity register is memory that bares the identity of the software. In the combination of Angelo and Arbaugh, Angelo discloses memory that contains a hash (Fig. 2) and therefore a type of identity because it is used to identity the software and Arbaugh discloses storing the public key certificates and therefore the public keys of the

executable code (section 3.1). In reference to the applicant's argument that there is no discussion whatsoever in Angelo of creation of OS identity: This is not persuasive. Arbaugh discloses calculating and storing the digital signature and certificate (operating system identity; 3.2.2). Arbaugh also discloses the verification of public key certificates (section 3.1 paragraph 3.4). Claim 19 is silent on what or who signs the OS certificate.

The applicant argues further, "an application program is not an operating system, and operating system is not an application program." This is not found persuasive because an operating system is software that is a program and an application program is a program. Although operating systems and application programs are programs with different functions they are both defined as programs.

Therefore, the examiner asserts that Angelo and Arbaugh do teach or suggest the subject matter broadly recited in independent Claims 1, 3, 11, and 19. Dependent Claims 6, 8, 14, 16, and 20-21 are also rejected at least by virtue of their dependency on independent claims and by other reason set forth in this office. Accordingly, rejections for claims 1-77 are respectfully maintained.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1-5, 7, 9-13, 15, and 17-19, are rejected under 35 U.S.C. 103(a) as being unpatentable over Angelo (5,944,82 1) in view of Arbaugh.

In reference to claim 1, Angelo discloses a system that comprises a central processing unit (CPU: part 100 Fig. 1 in combination with column 6 lines 8-13) and an operating system (OS), the CPU having a software identity register (Fig. 2 in combination with column 9 lines 35-38), a method for booting the operating system. The secure location is memory and therefore performs the same function as the register of the software identity register. Furthermore Angelo discloses setting the software identity register to a result of the computed hash value (Fig. 3 and Fig. 4).

Although Angelo discloses saving the hash value (identity of the program) in memory, Angelo does not expressly disclose computing a cryptographic function of at least a portion of the operating system and setting the software identity register to a result of the computed cryptographic function.

Arbaugh discloses a system that verifies the kernel (operating system) by calculating the cryptographic and storing the hash of the operating system level (page 4 section 3.2. 1 paragraph 2 in combination with section 3.2.2 paragraph 4). The cryptographic hash is the identity of the operating system since it is used to verify the integrity of the system.

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system as in Arbaugh in the system of Angelo. One of ordinary skill in the ali would have been motivated to do this because calculating the cryptographic hah function is used to calculate the integrity of a function a system

is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

In reference to claim 2, Angelo discloses further a method comprising defining a secure storage space, access to which is based in part on the result set in the software identity register (column 9 lines 12-25). The integrity of the huh table is verified by the table hash value stored in the SMM memory.

In reference to claims 3 and 11, Angelo discloses further a system that comprises a central processing unit (CPU: part 100 Fig. 1 in combination with column 6 lines 8-13) and an operating system (OS), the CPU having a software identity register (Fig. 2 in combination with column 9 lines 35-38). The software identity register is a register that stores the identity of related software. A register is a high-speed memory within a microprocessor used to hold data. Angelo discloses setting the software identity register to a result of the computed cryptographic function (Fig. 3 and Fig. 4). Angelo discloses further a system wherein in an event that the operation completes correctly, the software identity register (memory) contains the identity of the program (column 10 lines 16-28) and in an event that the operation fails to complete correctly, the software identity register contains a value other than the identity of the program; and examining a content of the software identity register to verify the identity of the program(column 10 lines 39-65). The hash value can be deleted; this would be setting the value to something other than the correct hash value. The user is also given a choice to update the value and put in a value that is different from the correct hash value.

However Angelo does not expressly disclose the identity of the software being the identity of the operating system.

Arbaugh discloses a system that verifies the kernel (operating system) by calculating the cryptographic hash of the operating system level (page 4 section 3.2. 1 paragraph 2 in combination with section 3.2.2 paragraph 4). The cryptographic hash is the identity of the operating system since it is used to verify the integrity of the operating system. The system of Arbaugh also expressly discloses a system for loading the operating system (Figure 3).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system as in Arbaugh in the system of Angelo. One of ordinary skill in the art would have been motivated to do this because calculating the cryptographic hash function is used to calculate the integrity of a function a system is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

In reference to claims 4, 9, 10, 12, 17, and 18, the identity comprises a public key of a correctly signed block of code from the operating system, and examining a content of the software identity register comprises verifying a signature of the signed block of code against the public key (Section 3.2.2 paragraph 2 Arbaugh).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system as in Arbaugh in the system of Angelo. One of ordinary skill in the art would have been motivated to do this because calculating the cryptographic hash function is used to calculate the integrity of a function a system is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

In reference to claims 7 and 15, that further comprises the authentication of additional blocks of code.

Arbaugh teaches authenticating sections of code using a signature (page 4 Section 3.2.2 paragraph 2).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system as in Arbaugh in the system of Angelo. One of ordinary skill in the art would have been motivated to do this because calculating the cryptographic hash function is used to calculate the integrity of a function a system is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

In reference to claim 19, Angelo teaches a system that includes a CPU (part 100 Fig. 1 in combination with column 6 lines 8-13) and an operating system (OS), the CPU having a software identity register (Fig. 2 in combination with column 9 lines 35-38). In addition, Angelo discloses a system wherein in an event that the operation completes correctly, the software identity register contains the identity of the operating system (column 10 lines 16-28) and in an event that the operation fails to complete correctly, the software identity register contains a value other than the identity of the operating system; and examining a content of the software identity register to verify the identity of the operating system (column 10 lines 39-65). The hash value can be deleted; this would be setting the value to something other than the correct hash value.
The user is also given a choice to update the value and put in a value that is different from the correct hash value.

However, Angelo does not expressly disclose having a pair of private and public keys and a software identity register that holds an identity of the operating system. The identity of the software created in Angelo is not expressly disclosed as the identity of the OS containing the and signing the OS certificate using the CPU private key.

Arbaugh discloses a system that verifies the kernel (operating system) by calculating the cryptographic hash of the operating system level (page 4 section 3.2. 1 paragraph 2 in combination with section 3.2.2 paragraph 4). The cryptographic hash is the identity of the operating system since it is used to verify the integrity of the operating system. Arbaugh also teaches the use of digital signatures and public key certification, therefore the use of private and public keys (page 4 section 3.2. 1 paragraph 1).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system as in Arbaugh in the system of Angelo. One of ordinary skill in the art would have been motivated to do this because calculating the cryptographic hash function is used to calculate the integrity of a function a system is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

In reference to claims 5 and 13 are rejected as in rejection for claims 3 and 11. Angelo discusses a hash value generated by an integrity assessment code that is specific to a given software application although the disclosed embodiment of the invention utilizes a hash table 206 containing hash values generated by a secure hash algorithm 208, it is contemplated that many types of modification detection codes could be utilized. Of importance to the invention is that each piece of software to be tracked has a corresponding and fairly unique value that

represents the unaltered state of the software, and that this value be stored in a secure memory location (Fig. 3).

Claims 22, 25-26, 30-38, 40-41, 43-54, 56-58, 69, 72-73, and 75-76 are rejected under 35 U.S.C. 103(a) as being unpatentable over Barr (6, 189,100 B1) in view of Arbaugh and Angelo (5,944,821).

In reference to claims 22, 25, 30, 31, 32, 34, 35, 38, 40, 43, 45, 48-54, 56-58, 69, 73, and 76, Barr suggests method for establishing a chain of trust between a subscriber unit and a content provider, the subscriber unit having a central processing unit (CPUI and an operating system (OS), the CPU having a pair of private and public keys (column 9 lines 10-24), a manufacturer certificate supplied by a manufacturer of the CPU (column 9 lines 50-55), and a software identity register that holds an identity of the operating system (column 9 lines 10-23), the method comprising: submitting a request from the subscriber unit to the content provider, the request specifying a particular content (Fig. 7A); generating, at the content provider, a challenge nonce (Fig. 7A); returning the challenge nonce from the content provider to the subscriber unit (Fig. 7A); forming, at the subscriber unit, an OS certificate containing the identity from the software identity register, information describing the operating system, the challenge nonce, and the CPU public key and signing the OS certificate using the CPU private key (column 9 lines 10-23); passing the OS certificate and the CPU manufacturer certificate from the subscriber unit to the content provider (column 9 lines 50-55); and evaluating, at the content provider, the OS certificate and the CPU manufacturer at the content provider to determine whether to reject or 11511 the request (column 9 lines 50-55 in combination with column 8 lines 17-24). The

examiner defines a software identity register as a register that stores the identity of related software. A register is a high-speed memory within a microprocessor used to hold data. Barr discloses a password that is encrypted and stored in a secure location (column 7 lines 59-67). The secure location is memory and therefore performs the same function of storing data as the register of the software identity register. The speed with which the data is to be retrieved depends on the design.

Barr does not expressly disclose the software identity being calculated and stored. Arbaugh discloses a system that verises the kernel (operating system) by calculating the cryptographic hash of the operating system level (page 4 section 3.2. 1 paragraph 2 in combination with section 3.2.2 paragraph 4). The cryptographic hash is the identity of the operating system since it is used to verify the integrity of the operating system. Arbaugh also teaches the use of digital signatures and public key certification, therefore the use of private and public keys (page 4 section 3.2. 1 paragraph 1).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system in Arbaugh in the system of Barr. One of ordinary skill in the art would have been motivated to do this because calculating the cryptographic hah function is used to calculate the integrity of a function a system is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

In addition, Angelo discloses a system wherein in an event that the identity of the operating system is stored in the form of a hash value in a hash table (column 10 lines 16-28).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to update the hash value as described by Angelo in the system described by Barr.

One of ordinary skill in the art would have been motivated to do this because trusted software may become vulnerable to attack and can no longer be relied upon to perform the trusted operation, recalculating the hash value and updating the huh table will revalidate the trusted software or reconfigure the integrity assessment (Angelo column 4 lines 17-24).

In reference to claim 26, Arbaugh discloses a system that verifies the kernel (operating system) by calculating the cryptographic hash of the operating system level (page 4 section 3.2. 1 paragraph 2 in combination with section 3.2.2 paragraph 4). The cryptographic hash is the identity of the operating system since it is used to verify the integrity of the operating system. Arbaugh also teaches the use of digital signatures and public key certification, therefore the use of private and public keys (page 4 section 3.2. 1 paragraph 1).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system as in Arbaugh in the system of Barr. One of ordinary skill in the art would have been motivated to do this because calculating the cryptographic hash function is used to calculate the integrity of a function a system is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

In reference to claim 36, the identity comprises a digital signature on a block of code from the operating system (column 6 lines 37-39).

In reference to claim 33, wherein forming a generator key and generating a storage key comprises creating a storage key SK using the formula SK = SHA (CPU-specific secret, OS-

specific data, seed). Angelo suggests the calculation of a hash value from a hash algorithm (Fig. 2 in combination with Fig. 3).

In reference to claim 37 is rejected as in rejection for claim 35.

Barr does not expressly disclose the operating system's identity comprising a hash digest of a block of code from the operating system, and examining a content of the software identity register comprises hashing the block of code.

Angelo discusses a hash value generated by an integrity assessment code that is specific to a given software application although the disclosed embodiment of the invention utilizes a hash table 206 containing hash values generated by a secure huh algorithm 208, it is contemplated that many types of modification detection codes could be utilized. Of importance to the invention is that each piece of software to be tracked has a corresponding and fairly unique value that represents the unaltered state of the software, and that this value be stored in a secure memory location (Fig. 3).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the hash value as an identity and examine the software integrity using the hash value as disclosed by Angelo in the system disclosed by Barr. One of ordinary skill in the art would have been motivated to do this because it is intended to be computationally infeasible to modify data so as to preserve a specific modification detection code value.

In reference to claims 41, 44, 46, 47, and 75, Angelo teaches a system that includes a CPU (part 100 Fig. 1 in combination with column 6 lines 8-13) and an operating system (OS), the CPU having a software identity register (Fig. 2 in combination with column 9 lines 35-38). In addition, Angelo discloses a system wherein in an event that the atomic operation completes

correctly, the software identity register contains the identity of the operating system (column 10 lines 16-28) and in an event that the operation fails to complete correctly, the software identity register contains a value other than the identity of the operating system; and examining a content of the software identity register to verify the identity of the operating system (column 10 lines 39-65). The hash value can be deleted; this would be setting the value to something other than the correct hash value. The user is also given a choice to update the value and put in a value that is different from the correct hash value.

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the hash value as an identity and examine the software integrity using the hash value as disclosed by Angelo in the system disclosed by Barr. One of ordinary skill in the art would have been motivated to do this because it is intended to be computationally infeasible to modify data so as to preserve a specific modification detection code value.

However, Angelo does not expressly disclose having a pair of private and public keys and a software identity register that holds an identity of the operating system. The identity of the software created in Angelo is not expressly disclosed is the identity of the OS containing and signing the OS certificate using the CPU private key.

Arbaugh discloses a system that verifies the kernel (operating system) by calculating the cryptographic hash of the operating system level (page 4 section 3.2. 1 paragraph 2 in combination with section 3.2.2 paragraph 4). The cryptographic hash is the identity of the operating system since it is used to velify the integrity of the operating system. Arbaugh also teaches the use of digital signatures and public key certification, therefore the use of private and public keys (page 4 section 3.2. 1 paragraph 1).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system ms in Arbaugh in the system of Barr. One of ordinary skill in the art would have been motivated to do this because calculating the cryptographic hash function is used to calculate the integrity of a function a system is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

In reference to claim 72, the identity comprises a public key of a correctly signed block of code from the operating system, and examining a content of the software identity register comprises verifying a signature of the signed block of code against the public key (Section 3.2.2 paragraph 2 Arbaugh).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to calculate the cryptographic hash of the operating system as in Arbaugh in the system of Bam One of ordinary skill in the art would have been motivated to do this because calculating the cryptographic hash function is used to calculate the integrity of a function a system is then said to posses integrity, without integrity no system can be made secure (Arbaugh Introduction).

Claim 6, 8, 14, 16, and 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Angelo and Arbaugh as applied to claims 3, 1 1, 19 are respectively above, and further in view of Sadowsky et al (6,230,285 B1).

In reference to claims 6, 8, 14, and 16, Angelo does not expressly disclose maintaining a boot log.

Sadowsky discloses maintaining a boot log (Fig 4). Further Sadowsky suggest the boot file comprising appending at least a portion of the identity to a boot log (column 4 lines 65 and 66).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to append the identity to the boot log of Sadowsky in the system of Angelo. One of ordinary skill in the art would have been motivated to do this because it will show the cause of boot failure (column 5 lines 12-15).

In reference to claims 21, the method wherein creating an identity of the OS comprises forming the OS certificate with one or more items from a boot log containing identities of software components that are executing on the CPU. The boot log discussed by Sadowsky contains information such as the device driver and executables (column 4 lines 65 and 66). This information is shared with the certificate information suggested by Barr.

Claim 23, 24, 39, 42, 55, 59-62, and 71 are rejected under 35 U.S.C. 1O3(a) as being unpatentable over Barr, Arbaugh, and Angelo as applied to claim 22, 35, and 56 respectively above, and further in view of Sadowsky et al (6,230,285 B1).

In reference to claims 24, 39, 42, 55, 55, 62, Barr does not expressly disclose maintaining a boot log.

Sadowsky discloses maintaining a boot log (Fig 4). Further Sadowsky suggest the boot file comprising appending at least a portion of the identity to a boot log (column 4 lines 65 and 66).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to append the identity to the boot log of Sadowsky in the system of Barr. One of ordinary skill in the art would have been motivated to do this because it will show the cause of boot failure (column 5 lines 12-15).

In reference to claims 23, and 71, the method wherein creating an identity of the OS comprises forming the OS certificate with one or more items from a boot log containing identities of software components that are executing on the CPU. The boot log discussed by Sadowsky contains information such œs the device driver and executables (column 4 lines 65 and 66). This information is shared with the certificate information suggested by Barr.

Claims 20 is rejected under 35 U.S.C. 103(a) as being unpatentable over Angelo and Arbaugh as applied to claims 19 above, and further in view of LeBourgeois (6,026,166).

LeBourgeois further suggests submitting the signed software identity register (the identity of the user) over a network to a third party to prove an identity of the operating system to the third party (Fig 3A and Fig. 3B).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to bind the identification of the device drive to the signature of the certificate as in LeBourgeois in the system of Angelo. One of ordinary skill in the art would have been motivated to do this because it is useful in ensuring that digital products are authorized for use on only one machine (column 3 lines 21-23).

Claims 63-68, 70, 74, and 77 are rejected under 35 U.S.C. 103(a) is being unpatentable over Barr, Arbaugh, and Angelo as applied to claims 56, 73, 69, and 76 above, and further in view of LeBourgeois (6,026,166).

In reference to claims 63 and 64, Barr does not expressly disclose the certificate containing the identities of the device drivers.

LeBourgeois discloses the digital certification method where the signature is dependent on the user identity (column 3 lines 54-57). In this case the user would be the device driver of the CPU.

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to bind the identification of the device drive to the signature of the certificate as in LeBourgeois in the system of Barr. One of ordinary skill in the art would have been motivated to do this because it is useful in ensuring that digital products are authorized for use on only one machine (column 3 lines 21-23).

In reference to claim 65 and 66, LeBourgeois further discloses submitting, by the user computer, a request to the third party (the certificate server) for access to specific content; evaluating, by the third party, whether to permit access based on the level of trust associated with the user computer (Fig. 3B).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to send the request to a certificate server for access to specific content as disclosed by LeBourgeois in the system of Barr. One of ordinary skill in the art would have been motivated to do this because the certificate server will prevent an imposter from creating a message purportedly from the original sender (column 1 lines 22-59)

In reference to claims 67 and 68, the access comprises transmitting, from the third party (the certificate server), a storage key for the specific content to the user computer through the secure connection (the connection between the merchant and the certificate server), wherein the specific content was previously stored on the user computer (Fig 3A and 3B). The specific content was obtained outside the secure connection (the user system; Fig. 3A).

In reference to claims 70, 74, and 77, LeBourgeois further suggests submitting the signed software identity register (the identity of the user) over a network to a third party to prove an identity of the operating system to the third party (Fig 3A and Fig. 3B).

Claims 27-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over Barr, Arbaugh, and Angelo as applied to claim 22 above, and further in view of Barlow et al (6, 038,551).

Barr discloses the use of certificates for the operating system, however does not expressly disclose the use of a manufacturing certificate.

Barlow discloses the use of a manufacturing certificate to verify the manufacturer and therefore whether to trust the manufacturer (column 8 line 66 to column 9 line 20).

At the time the invention was made, it would have been obvious to a person of ordinary skill in the art to compare the manufacturer certificate and the operating system certificate. One of ordinary skill in the art would have been motivated to do this because to prevent possible covert attacks from malicious software applications which attempt to gain unauthorized access to resources on the IC card (column 8 line 66 to column 9 line 3).

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Paula W Klimach whose telephone number is (571) 272-3854. The examiner can normally be reached on Mon to Thr 9:30 a.m to 5:30 p.m.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kim Vu can be reached on (571) 272-3859. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

PWK
Monday, February 07, 2005

KIM VU
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2